REMARKS

Applicant respectfully requests reconsideration of the present application, as amended, and consideration of the following remarks. No new matter is introduced by the amendments.

Oath/Declaration

A petition under 37 C.F.R. § 1.47(a) is being filed concurrently herewith.

Specification

The title suggested by the Examiner has been adopted. Furthermore, appropriate corrections have been made to specification based on the Examiner's comments in paragraph 4 and 5 of the Office Action under reply.

Drawings

The Examiner's objections to the drawings have been noted and in response appropriate changes have been made which are being submitted for approval under cover of a separate letter to the draftsperson.

Claim Rejections Under 35 U.S.C. § 102

The Examiner has rejected claims 1-18 under 35 U.S.C. §102 as being anticipated by Grochowski, et al. (U.S. Patent No. 5,535,347).

Claim 1 as amended recites a method for aligning an instruction stream comprising two elements namely (1) rotating data bytes of the instruction stream by a number corresponding to a byte length of a previous instruction in the instruction stream; and (2) shifting the data bytes to a start of the instruction immediately following the previous instruction based upon the byte length of the previous instruction.

An example according to one embodiment of the method recited in claim 1 is provided on page 14 of the specification and in pertinent part reads "Initially, rotator 504 contains bytes 2-

13 as shown in Table 2 for time 1. The shifter 506 contains bytes 2-13 and length is 0. LEN 402 determines the length of A. The length of A is returned to shifter 506. At time 2, shifter 506, using the length of A of 5 bytes, shifts bytes from rotator 504 into shifter 506 offset by the length of A and shifts A to length decode unit (LD) 508. Thus, rotator 504 contains bytes 2-13, the shifter 506 now contains bytes 7-13, and LEN 402 contains bytes 2-6 (instruction A). Shifter 506 then shifts bytes 7-13 to LEN 402 for length determination of instruction B. At time 3, shifter 506, using the length of B of 3 bytes, shifts instruction B into LEN 402, bytes 10-18 are shifted into shifter 506, and rotator contains bytes 7-18. The process is repeated in order to shift instruction C into LEN 402 as shown in Table 2."

Thus, for Grochowski, et al. to anticipate claim 1, Grochowski, et al. must teach or disclose a rotating step followed by a shifting step, each step being applied to data bytes of an instruction stream.

Analysis of Grochowski, et al. reveals that Grochowski, et al. teaches rotating a single bit of a 32 bit vector having a single bit set to indicate a byte position to indicate the length determined for an instruction (see columns 5 and 6). A significant difference between the rotation step recited in claim 1 and the rotation taught by Grochowski, et al. is that the Grochowski, et al. rotation is of a bit within a 32 bit vector, whereas the rotation as recited in claim 1 is of data bytes in an instruction stream. Further, Grochowski, et al. fails to teach the shifting step of claim 1. For at least these reasons, it is respectfully submitted that claim 1 is not anticipated by Grochowski, et al. Claims 2-10 depend on claim 1 and therefore include all limitations of claim 1. Thus, these claims are also not anticipated by Grochowski, et al.

Independent claim 11 and 13 include limitations which are similar to the limitations of claim 1 which have already been discussed above. Thus, on the basis of the arguments presented in respect of claim 1, it is respectfully submitted that claims 11 and 13 are not anticipated by Grochowski, et al. Claim 12 depends from claim 11, whereas claims 14-17 depend from claim 13 and, as such, claim 12 and claims 14-17 include all limitations of claim 11 and 13, respectively. Thus, it is submitted that claim 12 and claims 14-17 are also not anticipated by Grochowski, et al.

Rejections Under 35 U.S.C. § 103(a)

The Examiner has rejected claim 19 under 35 U.S.C. § 103(a) as being unpatentable over Grochowski, et al. (U.S. Patent No. 5,535,347).

In order to establish a prima facie of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on applicant's disclosure. In re Vaeck, 947 F.2d 488,20 USPQ2d 1438 (Federal Circuit 1991).

Claim 19, depends on claim 1 and therefore includes all limitations of that claim. As argued above Grochowski, et al. does not teach or suggest all limitations of claim 1 and therefore also of claim 19. Thus, claim 19 cannot be found to be obvious in view of Grochowski, et al.

It is respectfully submitted that in view of the amendments and remarks set forth herein, all rejections have been overcome. All pending claims are now in a condition for allowance, which is earnestly solicited. Authorization is hereby given to charge our deposit account 02-2666 for any charges that may be due. Furthermore, if an extension is required Applicant hereby requests such an extension.

Respectfully submitted,
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: 7/15/02

Michael J. Mallie
Reg. No. 36,591

12400 Wilshire Blvd.
Seventh Floor
Los Angeles, CA 90025-1026
(408) 720-8300

OIPE JC67 JUL 22 2002 PATENT & TRADEMARK OFFICE

MARKED VERSION SHOWING CHANGES MADE

IN THE TITLE:

Please amend the title to read as follows:

METHOD AND SYSTEM FOR A TWO STAGE PIPELINED INSTRUCTION DECODE AND ALIGNMENT USING PREVIOUS INSTRUCTION LENGTH

IN THE SPECIFICATION

On page 6, please amend the paragraph starting at line 19 through line 4 of page 7 to read as follows:

As defined within the Intel architecture instruction set, an exemplary macro-instruction 100 may comprise [zero to fourteen] instruction prefixes 102 (each instruction prefix 102 being 0-4 [one byte] bytes in length), an opcode 104 (1-2 bytes in length), a ModR/M operand indicator 106 (0-1 byte in length), an SIB of 108 (0-1 lengths in byte), address displacement 110 ([0-4] 0, 1 or 4 bytes in length), and an intermediate data constant 112 ([0-4] 0, 1 or 4 bytes in length). Opcode 104 may be either one or two bytes in length. For two-byte opcodes, the first byte is 0F.

On page 10, please amend the paragraph starting at line 5 through line 13 to read as follows:

In the second pipe stage, instruction length decoder (ILD) 308 determines the length of the current instruction. IA instructions are variable length instructions varying in length from 1 to 15 bytes with prefixes and 1 to 11 bytes without prefixes. In order to properly align and decode the instructions, the length of the instruction must be determined. The bytes that are received from the ALN 306 stage are assumed to start with the first byte of instruction. The ILD

308 decodes these instruction bytes, determines the length of the instruction, and sends the length to the ALN 306 for subsequent instruction realignment and to [the DE1] a decode stage for marking the instruction boundaries.

On page 12, please amend the paragraph starting at line 8 through page 13, line 2 to read as follows:

Figure 5 is a block diagram showing architecture details of ALN 306 and LEN 402. Data stream bytes are received into two 10x16-byte buffers [306] 304 from MIQ buffers 302. The data stream is rotated into rotator 504. In one embodiment, rotator 504 consists of 12 bytes. Rotator 504 rotates the data bytes of two instructions. In one embodiment, an instruction has a maximum length of 11 bytes (without prefixes). If 12 bytes captures both instructions, then ALN 306 will have a maximum throughput. Rotator 504 is one pipe stage behind the decoding of the current instruction. Using the length vector obtained from the LEN 402, ALN 306 next shifts the current instruction into shifter 506. Shifter 506 shifts to the exact instruction start based on the length of the first instruction in the data stream. If rotator 504 does not contain the entire instruction required by shifter 506, rotator 504 rotates instruction data from buffers 304. Shifter 506 output gives the current instruction for the current pipe stage. It is assumed that the first instruction within the data stream begins at the beginning of the data buffer. Thus, during the current pipe stage, rotator 504 is obtaining instruction data for the current instruction while shifter 506 is obtaining data for the next instruction in the data stream.

On page 14, please amend the paragraph starting at line 3 through line 13 to read as follows:

Initially, rotator 504 contains bytes 2-13 as shown in Table 2 for time 1. The shifter 506 contains bytes 2-13 and length is 0. LEN 402 determines the length of A. The length of A is returned to shifter 506. At time 2, shifter 506, using the length of A of 5 bytes, shifts bytes from

rotator 504 into shifter 506 offset by the length of A and shifts A to LEN [402] length decode unit (LD) 508. Thus, rotator 504 contains bytes 2-13, the shifter 506 now contains bytes 7-13, and LEN 402 contains bytes 2-6 (instruction A). Shifter 506 then shifts bytes 7-13 to LEN 402 for length determination of instruction B. At time 3, shifter 506, using the length of B of 3 bytes, shifts instruction B into LEN 402, bytes 10-18 are shifted into shifter 506, and rotator contains bytes 7-18. The process is repeated in order to shift instruction C into LEN 402 as shown in Table 2.

On page 16, please amend the paragraph starting at line 1 through line 10 to read as follows:

Figure 7 is a block diagram showing architectural details of one embodiment of the [instruction] length [decoder] decode unit (LD) 508. LD 508 determines the length of various portions of the instruction received from ALN 306. ALN 306 shifts the current instruction from the shifter 506 onto the LD 508. Within the [LD] 508, opcode-plus-immediate logic unit OPIMM 602 determines the length of the opcode 104 and immediate data 112 of the current instruction. B0 and B1 are inputs to OPIMM 602 together with the operand-size (Osz) signal. The Osz signal selects the sizes of operands that instructions operate on. When the 16-bit Osz signal is in force, operands may be either 8 or 16 bits. When the 32-bit Osz signal is in force, operands may be 8 or 32 bits.

On page 18, please amend the paragraph starting at line 9 through line 14 to read as follows:

Table 3 shows the possible outputs from OPIMM 602. The outputs are dependent on whether the opcode 104 is one or two bytes and the possible lengths of the immediate data 112. The immediate data may be 1, 2, 4 or 6 bytes in length. Thus, the opcode plus immediate may be

1, 2, [3, 4, 5, 6 or 7] or 6 bytes in length. Table 3 indicates the possible combinations of opcode and immediate displacement.

IN THE CLAIMS:

Please amend the claims as follows:

1. (Twice Amended) A method for aligning an instruction stream comprising:

rotating data bytes of the instruction stream by a number corresponding to a byte length of a previous instruction in the instruction stream; and

shifting the data bytes to a start of the instruction immediately following the previous instruction based upon [a] the byte length of [an immediately prior macro] previous the instruction.

3. (Twice Amended) The method of claim 1 further comprising:

receiving [a] the byte length of the [an immediately prior] previous instruction from a length decode logic unit.

5. (Twice Amended) The method of claim 1 wherein said shifting shifts to an exact start of the instruction immediately following the previous instruction.

7. (Twice Amended) Logic for aligning an instruction stream comprising:

a rotator logic unit for rotating data bytes of the instruction stream by a number corresponding to a byte length of a previous instruction in the instruction stream;

a shifter logic unit for shifting the data bytes to a start of the instruction immediately following the previous instruction based upon [a] the byte length of [an immediately prior] the previous instruction.

9. (Twice Amended) The logic of claim 7 further comprising:

a length vector [for] providing the byte length of [an immediately prior] the previous instruction.

11. (Twice Amended) A processor to align an instruction stream comprising:

a rotator logic unit for rotating data bytes of the instruction stream by a number corresponding to a byte length of a previous instruction ;

a shifter logic unit for shifting the data bytes to a start of the instruction immediately following the previous instruction based upon [a] the length of [an immediately prior] the previous instruction.

12. (Twice Amended) The processor of claim 11 further comprising:

a length vector [for] providing the byte length of [an immediately prior] previous instruction.

13. (Twice Amended) A system for aligning an instruction stream comprising:

means for rotating data bytes of the instruction stream by a number corresponding to a byte length of a previous instruction in the instruction stream; and

means for shifting the data bytes to a start of the instruction immediately following the previous instruction based upon [a] the byte length of [an immediately prior] the previous instruction.

14. (Once Amended) The method of claim 1 further comprising:

determining [a] the byte length of [a current instruction] the previous instruction.

15. (Once Amended) The method of claim 14 wherein the byte length of the [current] previous instruction is based upon a length of an opcode and a length of immediate data.

16. (Once Amended) The method of claim 14 further comprising:

determining if an opcode extension byte is required to determine the byte length of the [current] previous instruction.